

第4章

ARMプロセッサでレーザ・ディスプレイ装置を実現する

——本誌2006年3月号付属ARM7基板による実時間制御応用事例

竹内誠二

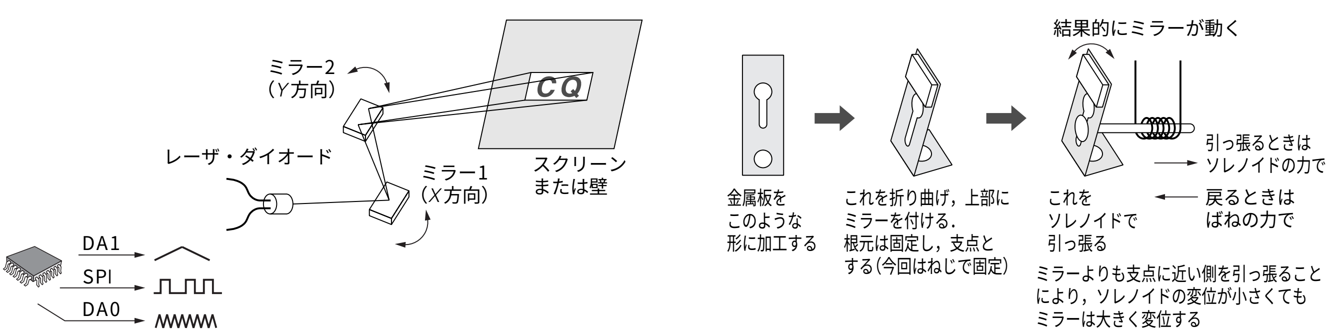
ここでは，本誌2006年3月号に付属したARM7基板を利用してレーザ・ディスプレイ装置を試作した例を紹介する．レーザ光を2枚のミラーで制御し，ブラウン管のように走査することで画像表示を実現している．アクチュエータは，ソレノイド（電磁石）とばねを組み合わせて作成した．なお，本稿で解説したプログラムやデータは，本誌付属のCD-ROMに収録されている．（編集部）

本稿では，本誌2006年3月号に付属したARM7基板（マイコン・ボード）を使い，ビットマップを出力するレーザ・ディスプレイ装置を試作した例を紹介します．「電子工作」というよりは「機械工作」になってしまいましたが，たまにはメカを動かしてみるのもおもしろいのではないでしょうか．

● レーザ光でスクリーンに絵や文字を描く

レーザ・ディスプレイ装置とは，レーザを用いて映像表現を行うものをいいます．ライブ会場やクラブなどで使用されていたりするので，目にしたことがある方は多いと思います．表現のしかたは大きく分けて，空間を貫くレーザ光そのものを見せる方法と，レーザ光によって壁やスクリーンに絵や文字を描く方法の2種類があります．今回試作したのは後者です．

ここで「絵」を描く方式は2通り考えられます．まず一つ目は，レーザ光線そのものを任意の方向に動かし，ひと筆書きのように描いてしまう方式です．もう一つは，投影面全体を走査し，それに合わせてレーザ光をON/OFFさせて描く方式です（ブラウン管の表示方法と同じ）．今回は，後者を採用しました．理由は，後者の方式において，フレ



（a）レーザ・ディスプレイのシステム・イメージ　（b）ミラー機構

図1　レーザ・ディスプレイ装置

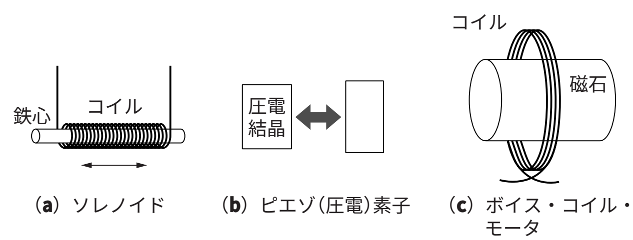
（a）のように，レーザ・ダイオードの光を二つのミラーで*X-Y*方向に振り，内蔵RAM上のビットマップを壁面に描き出す．ミラーの制御はマイコンのD-Aコンバータ出力で行い，レーザ・ダイオードはSPIでON/OFF制御を行う．D-Aコンバータからは三角波，SPIからは内蔵RAM上に展開したビットマップを送出する．（b）のように金属板を鍵穴状に加工した．下部を固定し，上部にミラーを付け，中央部をソレノイドで引っ張る．かぎ穴状に加工したのは，今回使用したソレノイドの鉄心の先端がT型に加工してあったため，引っかけるのにちょうどよかったから．ソレノイドに電流を印加することにより吸引され，金属板ばねの力で復帰する．

KeyWord　レーザ・ディスプレイ，アクチュエータ，ソレノイド，SPI，NJM2073，ARM7，ピエゾ素子，ボイス・コイル・モータ，ビットマップ，Thumb

Column アクチュエータについて

今回検討したアクチュエータは，**図A-1**の3種類です．それぞれ簡単に説明します．いずれの方法でも実現できると思います．これ以外にもさまざまなアクチュエータがあります．たまには，標準のモータではないものを動かしてみるのも，おもしろいかと思います．

- **ソレノイド** ── 簡単にいえば電磁石．コイルに電流を流し，鉄心を吸着する．なお，今回のように高速に往復運動をさせるためのものではなく，もっと低速で使用することが多い．通常はシャッタやバルブの開閉などに用いられる．また，復帰機構を内蔵しているものなどもある．
- **ピエゾ（圧電）素子** ── 圧電効果を持つ結晶（圧電結晶）を利用した素子．圧電結晶とは，圧力をかけると電圧が発生する結晶を指す．逆に，この結晶に電圧をかけると結晶内に圧力が発生し，伸縮する（縦横比が変わる）ため，往復運動を実現することが可能．変位が小さいため，実際に用いるときは金属板の裏にはるなどして変位を大きくする．身近な例では，圧電ブザーやプリンタ・ヘッドに使用されている．ちなみに，水晶も圧電結晶の一種である．
- **ボイス・コイル・モータ** ── 磁界中にコイルを配置し，これに対して電流を加え，フレミングの左手の法則により力を得る．身近な例では，スピーカの駆動機構がこれにあたる．また，ハード・ディスク装置（HDD）のヘッドなどにも使用されている．



図A-1　今回検討したアクチュエータ
(a)のソレノイドはコイルを電磁石として用い，鉄心を吸着する．(b)のピエゾ素子は電圧を加えることにより，結晶が変形することを利用する．(c)のボイス・コイル・モータは，磁界中に置いたコイルに電流を流し，発生する電磁気力を利用する．

ーム・メモリ上のビットマップを出力できるようにしてしまえば，絵の交換やスクロール，アニメーションなどを，プログラムしだいで自由に実現できるからです．

● マイコンのD-Aコンバータから三角波を出力して制御

まず第一に，秋葉原やホームセンタなどで簡単に入手できる材料によって，簡単なビットマップを表現できることを目標としました．ライブハウスなどで使われているプロ仕様と比べると見劣りしますが，自分の部屋の中で楽しむには十分だと思います．

方式は至ってシンプルです．レーザ光を面的に走査し，それに合わせてレーザ光をON/OFFすることにより，ビットマップを壁に描き出します（**図1(a)**）．これを実現する方法については，以下で一つ一つ具体的に説明していきます．

レーザ光を走査するには，ARM7基板に搭載されているマイコン（米国Analog Devices社の「ADuC7026」）が内蔵するD-Aコンバータを用います．まず，D-Aコンバータから「三角波」を出力します．これを何らかのアクチュエータ（駆動機構）に入力し，ミラー（鏡）を振ります．これにレーザ光を当てると，ミラーが振れているため，レーザ光線の向きも振れます．これを二つ用いることにより，*X-Y*方向に走査します．一般的なブラウン管のように「ノコギリ波」でないのは，追従させるものが電子線ではなく，無視できない質量を持つミラーだからです．

● 電磁石とばねで往復運動を実現

今回，アクチュエータとして，ソレノイドを採用しました．本来はボイス・コイル・モータなどのほうがよかったと思うのですが，ソレノイドが簡単に入手できたので，試しにそれで作ってみました．もちろん，ボイス・コイル・モータでも作成できますし，ピエゾ（圧電）素子でも作成できると思います．もしみなさんが作成するときは，入手しやすいものを使用すればよいと思います．

ソレノイドは，簡単に言えば電磁石です（上掲のコラム「アクチュエータについて」を参照）．構造としては，中空のコイルの中に鉄心が入っており，電流を流すとこれを吸着します．電流を切れば吸着力はなくなります．ただそれだけですが，吸着力がなくなっても元のところには戻ってこないため，元のところに戻すための何らかの機構が必要です．今回は，金属板でばね機構を作成しました（**図1(b)**）．この金属板の上にミラーをつけ，根元を固定し，てこの原理によって変位を増幅させることをねらっています．この金属板をソレノイドで引っ張り，ミラーを動かします．電流を切れば，金属板のばねの力により元に戻るというしかけです．

なお，一般に光線を走査する場合，ポリゴン・プリズムが使用されます．例えば，6角形のプリズムをモータで回転させ，これにレーザ光を入射すると，光線を「ノコギリ波」状に走査できます．こちらは単なる回転運動ですから，

ソレノイドによる往復運動よりも簡単に制御できるはずです．しかし，今回は真ん中に穴の空いたプリズムを調達できず，個人で加工するのも難しいと思ったので，この方法は断念しました．この結果，やや制御性は悪いと思われますが，単純な「ソレノイド＋ばね」の機構で往復運動を実現することにしました．

● レーザ光のON/OFF制御にはSPIを利用

さて，上記の方式でレーザ光を面的に走査できるようになりました．次は，これに合わせてレーザ光をON/OFFする機構を考えます．

今回はSPI（serial peripheral interface）からの出力を用いました．このようなON/OFF制御を行う場合，一般にはGPIO（general purpose input/output port）を用いますが，これを使用したのでは時間的に間に合いません．およそ常識的なSPIの使いかたではありませんが，内蔵する周辺モジュールを「出力したい波形を作る機構」と割り切り，今回はいちばんしっくりくるSPIを使用しました．SPIはシリアル通信モジュールで，送信データをビットごとに‘0’，‘1’として，一定の速度で送信します．2値のビットマップを内蔵RAM上に持った場合，これをそのまま送信できるので，今回の用途には非常につごうが良いのです．

● D-Aチャネルの同期にバッファ・モードを使用せず

それでは，さらに詳細な設計について説明します．

まずクロックですが，最高速のモードを用います（POWCONレジスタのCD＝0）．この速度でないと制御が間に合いません．

三角波は，D-Aチャネル0，1（DA0，DA1）から出力します．ここで，D-Aチャネル0はX方向，D-Aチャネル1はY方向としています．正確な三角波を出力するため，タイマ0を使用してタイミングを発生させます．そして，タイマ0の割り込みルーチンの中でD-A出力の値を書き換えます．

本マイコンのD-Aコンバータはバッファ・モードとアンバッファ・モードを持っています．複数のD-Aチャネルの同期をとるためにはバッファ・モードを使うのが正しいのですが，今回は使用していません．理由は，2チャネルなので，アンバッファ・モードでも誤差は1CPU時間程度で済むためです．また，レーザ出力のON/OFFを行っているSPIと同期をとりやすくするため，という意味もあります．ただし，誤差を最小限に抑えるために，あらかじめ次の値を先の回に計算しておき，割り込みの先頭で連続して値を書き込みます．

● Windowsのビットマップ形式にならって描画

ここで，描画方式について説明します．レーザのON/OFF制御はSPIの出力から直接行い，‘1’出力でON，‘0’出力でOFFとなっています．今回，実際には256ビット×192ビットの白黒のビットマップを内蔵RAM上に持っていますが，ここでは話を簡単にするため，32ビット×4ビットとして説明します（**図2**）．

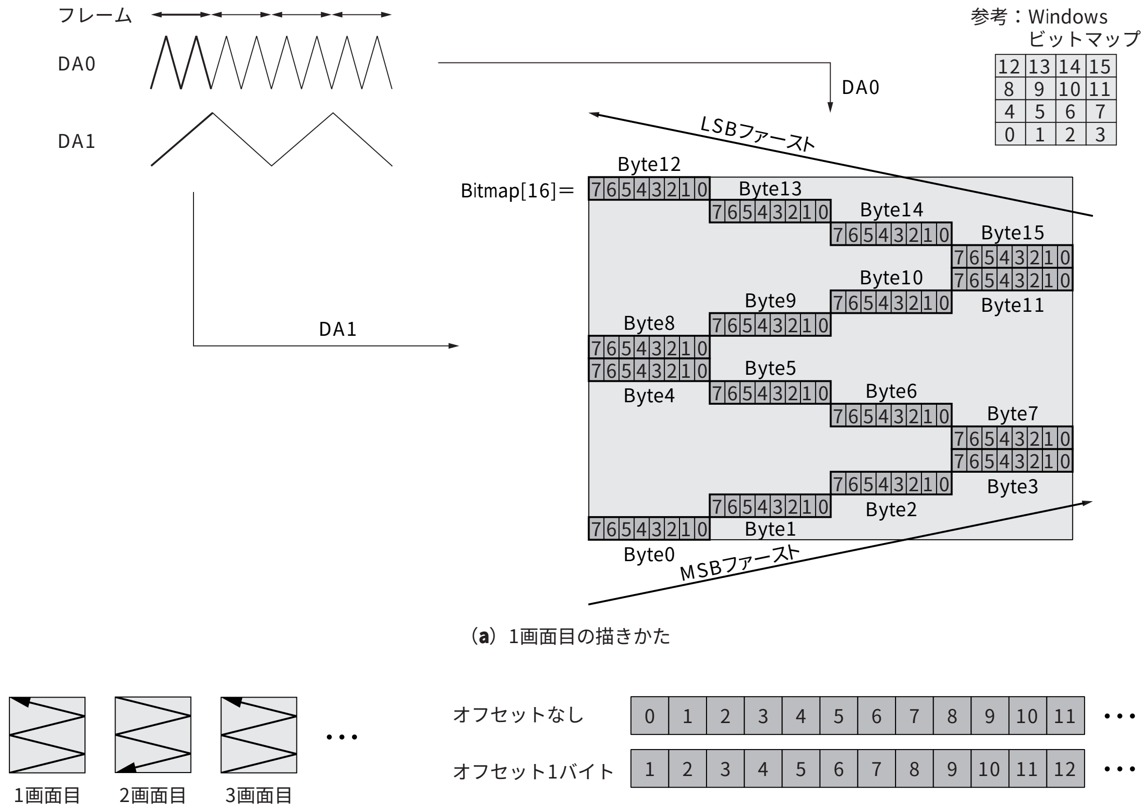
このビットマップは，Windowsのビットマップをそのままの形で持っており，左下から右下，そして上へとビットマップが格納されています．これを，D-Aチャネルから出力されている三角波に合わせて描画するために，まず最初の行はMSB（most significant bit）ファーストでByte 0～3を出力します．次の行は反対方向に描画するため，LSB（least significant bit）ファーストでByte 7～4の順に出力します．同じように，引き続きMSBファーストでByte 8～11，LSBファーストでByte 15～12の順で出力します．これで，1画面目の描画が完了します．

続いて，今度は逆方向に2画面目を描画します．この場合，MSBファーストでByte 12～15，LSBファーストでByte 11～8，MSBファーストでByte 4～7，LSBファーストでByte 3～0の順になります．これで振り出しに戻ることになるので，3画面目以降は先の説明と同じことの繰り返しとなります．したがって，プログラムは2画面分ずつ無限ループに入ることになります．

● プログラム上でオフセットを設定可能に

実際のプログラムについて説明します．タイマ0の割り込みを受け，D-Aチャネル0，1の出力値を書き換えます．その際に，SPIからの出力1バイト分も書き換えます．このため，8ビット分ずつまとめて処理していることになります．これはSPIの送信単位がバイト単位であるため，しかたがありません．したがって，実際のD-A出力波形は**図2**に示したとおり，バイトごとの階段状となりますが，ミラーには質量があるため，平滑化されると期待できます．

SPIの送信速度は，次のタイマ割り込みが発生するときにちょうど送信完了しているように設定しておきます．これにより，送信完了ステータスを確認せず，送信レジスタ（SPITX）にいきなり書き込んでしまっても問題ありません．



（**a**）1画面目の描きかた

（**b**）1画面目以降の描きかた

（**c**）オフセット

図2　描画方式の詳細

D-A チャネルから出力される三角波に合わせて，MSB ファーストまたはLSB ファーストを切り替えながら，SPI 出力のデータの描画を行う．往復で描画するため，第1フレームはD-A チャネル1の三角波の半波長のところで完了し，反対方向に第2フレームを描く．これを繰り返すことにより，描画を行う．D-A チャネルとSPI の間の遅延にはオフセットで対応する．

データは送信レジスタ書き換え後，すぐに送信されます．ただこの場合，「D-A チャネルのレジスタ書き換え～D-A出力の決定」の時間は標準（typ）で15μsですが，「送信レジスタの書き込み～実際のピン出力」の遅延時間は規定されていないので，結果的にD-A 出力とSPI 出力がどれだけずれるのかは不明です．また，ミラーやソレノイド機構は無視できない質量を持っており，金属ばねが三角波に100%追従できるとは限りません．これらに対応するため，プログラム上でオフセットを指定できるようにしています．通常，これらの規定されていない値も，まったく同じ条件（プログラム，質量）であれば，まったく同じタイミングのずれとなって現れると期待されるので，これで対応できるはずです．

以上の機構により，描画が行われます．ビットマップはUART によりパソコンからダウンロードします．上記のオフセットなどの調整も同じようにUART 経由で行います．

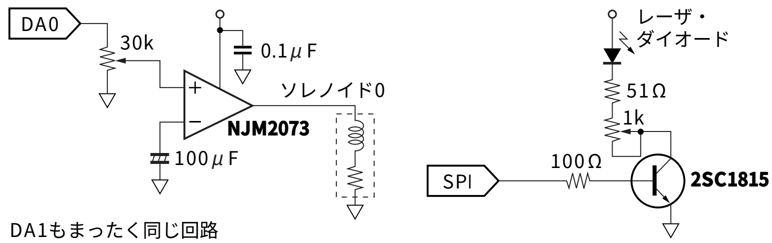
● パワー・アンプで駆動力とゲインの両方を得る

基板上の回路（**図3**）について説明します．

UART はARM7 基板に出ている端子を使用しています．本マイコンには，シリアル経由でプログラムをフラッシュ・メモリに書き込む機能がありますが，この機能を使用するときの結線と同じです．当然，同じコネクタをそのまま使用できます．

D-A コンバータの出力は，パワー・アンプ（NJM2073，スピーカ用）を用いて5V に増幅します．これは，ソレノイドが実際に電流を必要とするからです．OP アンプで増幅した場合，OP アンプの電流駆動能力が十分ではなく，ソレノイドが十分に動作しませんでした．トランジスタでエミッタ・フォロワの回路を組んだ場合，駆動力はまあまあったのですが，ゲイン（利得）が1倍しかとれないため，5V フルスイングを実現するためには2段以上の回路を組む必要があります．これに対して，今回用いたパワー・アン





（a）ミラー制御回路

（b）レーザ出力のON/OFF回路

部　品	型　名	個　数
パワー・アンプ	NJM2073	1
トランジスタ	2SC1815	1
レーザ・ダイオード	LM-102-B	1
可変抵抗　30Ω	―	2
1kΩ（光量調整用）	―	1
固定抵抗　51Ω	―	1
100Ω	―	1
電解コンデンサ　100μF	―	2
セラミック・コンデンサ 0.1μF	―	1
ソレノイド	SK0315A06AA	2

（c）部品リスト

図3　回路図

D-A出力を2個入りパワー・アンプNJM2073で増幅し，ソレノイドに与える．レーザの出力は2SC1815でスイッチする．

プは，1個で駆動力もゲインも得ることができるので，お手軽です．音響機器用のものですが，今回使用する三角波の周波数帯は可聴領域であり，スピーカそのものも「コーン」という質量体を動かしているので，それなりの駆動力を期待できると考えました．

なお，ソレノイドそのものは高電圧を印加しても問題なく，パワーを得るためにはこの方法もありますが，できるだけ単純に作成するために，一般のパワー・アンプを用い，単一電源システムで製作しました．

SPIの出力は，いったんトランジスタで受け，スイッチ動作させています．これは電流駆動能力を上げるためです．今回使用したレーザ・モジュールはクラス1，定格電流は40mA，入力信号SPIのビット・レートはおよそ400kbpsなので，トランジスタを一つ入れた回路にしました．レーザにはLEDと同じように電圧降下があります．電流制限用の抵抗はそれぞれに合わせてください．なお，レーザ光で描画した絵の写真を撮るつもりであれば，ボリュームを入れることをお勧めします．

今回，レーザ・ダイオードにはモジュール化されているものを使用しました．これは，裸のレーザ・ダイオード（半導体レーザ）に発振安定回路が付いているものです．このため，速度応答性が悪くなるはずですが，今回はとくに問題は起きませんでした．

● タイマ割り込みの処理の良否を「音」で判定

今回のプログラムでは，「音」が発生します．これは，機械的な振動を発生させているからです．ここで，タイマ割り込みに対して処理が間に合っている場合，D-Aコンバータの出力周波数は一定であり，一定の「音」がミラー機構から出力されます．ところが，タイマ割り込みに間に合わない場合，「うなり」が出力されます．これは，タイマ割り込みに処理が間に合わなくなってくると，処理が飛ばされ，結果的にその部分だけ間延びして周波数が落ちるからです．これを利用して，「音」を聞いていればタイマが間に合っているかどうかを判定することができます．もちろん，オシロスコープで確認することも可能ですが….

プログラム上のくふうですが，まず，割り込みルーチンを極力軽くしています．当初は割り算を使用したルーチンを書いていたのですが，まったく間に合わないため，すべてシフトを使った記述に書き換えました．また，多少見にくくなっても，演算量を減らすようにくふうしています．プログラム中では，この割り込みルーチンはタイマ0で起動されています．タイマ0にはHCLK（CPUと同じクロック）を指定しています．

カウント・ダウン値はUART経由で変更でき，間に合わなくなり始めると「うなり」が生じます．また，UARTの受信はメイン・ルーチン内でポーリングしており，そこからエコーバックさせているため，まったく間に合わなくなってくると反応しなくなります．このため，どこまでなら間に合うかを簡単にチェックできます．

今回のプログラムでは，800サイクルくらいで怪しくなり，620サイクルで反応しなくなります．たったこれだけのプログラムを動かすのに800サイクルかかるというのはちょっと遅い気もします．ただ，今回のマイコンはA-Dコンバータや D-Aコンバータが潤沢にあり，どちらかというとセンサ機器向けなので，今回のような実時間制御にはあまり向いていないのかもしれません．

● 割り込みルーチンに2段飛びで入る

今回，割り込みルーチンには2段飛びで入るようにしています．ARMプロセッサは割り込み時に32ビットのモードに戻ります．本マイコンのプログラム格納領域（内蔵フラッシュ・メモリ）は，CPUと16ビット幅で接続されているため，32ビット・コード（ARMコード）ではプログラム・フェッチに最低2サイクルかかることになり，性能が上がりません．本文に16ビット・コード（Thumbコード）の使用を指定しているとき，割り込みルーチン内も自動的に16ビット・コードにしてくれることを期待したのですが，そうはなっていませんでした．

これを改善するためにあれこれ試行錯誤した結果，このように2段飛びにすると，今回使用したKeil Software社（現在はARM社の傘下）の開発環境は16ビット・コードを出力してくれました．ただし，古いバージョンだと，この「2段飛び」にしたとき，レジスタの待避をまちがえてしまい，暴走します．最新バージョンでは，この点は修正されています．このように開発環境を使用するときは，つねに新しいバージョンが出ているかどうかをチェックするように心がけてください．

Keil社の開発環境にはエミュレータ機能やロジック・アナライザ機能が付属しているので，ほとんどこの上でプログラムをデバッグできます．実際にエミュレータ機能の結果を100％信じてよいかどうかは疑わしいのですが，それでもパソコン上で波形を確認できるというのは，たいへんに便利な機能です（pp.56-57のコラム「開発環境について」を参照）．

実を言うと，割り込みルーチンの最後でフラグをクリアする場合，確かに上記のように処理が飛ばされ，「うなり」が生じます．ただ，今回のプログラムでは先頭でフラグをクリアしているため，処理が飛ばされることはありません．本来は，いきなり反応しなくなるのが正しいのですが，実際にうなりは発生しています．これは，何らかの理由により，「割り込みルーチンから復帰直後に再び割り込みがかかる」という無限ループの中で，何かサイクルのずれを発生する要因があるということです．これは何かというと，メイン・ルーチンしかありません．少なくともこのマイコンは，割り込みから復帰したときに数命令動作させてからでないと，次の割り込みに入れないのだと思います．

これは，悪く言えば「割り込み応答性が悪い」ということですが，そもそも実時間制御を行うとき，CPUは十分にアイドル（待機）時間を持っていなければならないので，現実的には問題とはならないはずです．ただし，用途によっては，注意すべき項目になるかもしれません．

● 重要なのは変位の「大きさ」よりも「線形性」

プログラム上で三角波を発生させ，ディスプレイ装置（**写真1**）に入力すると，発振を始めます．ただし，ソレノイドはその構造から，吸着すればするほどその吸着力が大きくなり，非線形的に増加していきます．一方，ばねの反発力は，変位に対して線形です（$F = kx$）．したがって，ソレノイドとばねの距離は非常に微妙なものとなります．近すぎるとばねがソレノイドに吸着されてしまって，ぴくりとも動きません．遠すぎるとソレノイドの吸着力が小さくて，変位も小さくなってしまいます．

そこで，今回はねじで微調整できるようにしました．使用したねじはM2のサイズです．これはねじのピッチが0.4mmであるため，1回転で0.4mm動かすことができます．これにより，0.05mm（＝1/8回転）くらいの精度で調整で

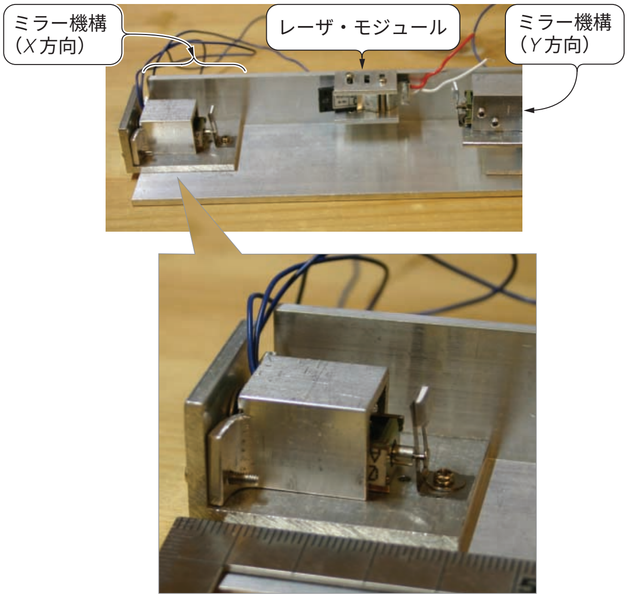


写真1　ディスプレイ部

上の写真の中央にあるレーザ・モジュールからレーザ光を出力し，左側のミラー機構で反射させ，これをさらに右側のミラー機構で反射させて描画する．左側のミラー機構は高速方向（X方向）であり，金属ばねが短辺方向に振動する．一方，右側のミラー機構は低速方向（Y方向）であり，金属ばねが長辺方向に振動する．このため，一見すると同じ向きになっているが，X-Y方向に光線を走査し，描画できる．なお，ミラー機構は左右で一部作りが異なっているが，これは試行錯誤の結果である．また，右側のミラー機構の下に台が入っているのは，高さを合わせるため．
下の写真のように，光学系は，ソレノイドを覆う形のハウジングに（向こう側から）ねじで固定されている．これを後ろ板側からばねで押し，ねじで引っ張ることで，台に固定されている金属ばねとの間隔を調整する．

きるようになります(最初は両面テープで調整していたが，だんだんはがれてきたので，ねじを使うことにした).

ただし，レーザ・ディスプレイについては，ある程度の変位が得られればよいということを覚えておいてください．この装置では変位は光線の角度として現れるため，装置とスクリーンの距離を離せば離すほど，見た目の像の大きさは拡大します．また，光源はレーザなので，レンズによって像の大きさを拡大・縮小できます．

上記変位の「大きさ」よりも重要なのは，変位の「線形性」です．今回は有限の質量を三角波で励振するというものなので，実際にきちんと応答してくれるかどうかわかりませんでした．実際に試す前は，「スピーカだって同じ範囲でやっているのだから，なんとかなるはず」と考えていました．結果的には，まったく問題なく追従しているようです．例えばD-A出力をテーブル出力に切り替えて，正弦波を入れるなどの補正が必要かと思っていましたが，結果的にはなんの補正もしていません．この範囲であれば，ほぼ問題なく追従するようです．

ただし，オフセットは発生します．それがSPIとD-A出力のずれによるものか，あるいは機構の質量によるものか，区別できません．とにかく適切なオフセットを入れてやれば像が安定するので，一定値であることは確かです．

● ばねをいくつか取り替えて追従性を改善する

なお，高速方向については，**図1(b)**に示した方向(ばねの長辺方向)ではなく，短辺方向に振動するものも利用できます．今回の機構では，使用したソレノイドの鉄心の頭がT型になっていることを利用して，かぎ穴型の加工を行い，ここに引っかけて使用しています．このときに，片側のみがばねに当たるようにして，かつ十分高速であれば，長辺方向の振動は現れず，短辺方向の振動のみとなるようです．

この場合，ソレノイドそのものも振動に追いつけないのか，ソレノイドとばねの間隔はかなり適当でもよくなるのですが，振動はしており，レーザ光を走査する分には問題

Column 開発環境について

今回，Keil Software社の開発環境を使用しました．これはソフトウェアの統合開発環境です．通常の開発ツールのように変数を見るためのウォッチ・ウィンドウなども装備していますが，ここで取り上げたいのはロジック・アナライザ機能と周辺モジュールのエミュレーション機能です．

この環境では，デバッグ・セッションにおいて，周辺モジュールのエミュレーションを行ってくれます．また，それらを含めた波形も表示できます(**図B-1**)．波形というと，ソフトウェア・プログラマなどにはなじみが薄いもので，筆者も最初はとまどいました．波形は縦軸に値を，横軸に時間を取って，変数の値を表示したものです．通常のソフトウェアでは「時間」というものを意識することがないため，ブレークポイントを設定し，そのつど値を見るのが普通です．しかし，今回のような組み込みソフトウェアにおいては，「実時間」というものが非常に重要になってきます．実際に見てみると，D-A0とD-A1から三角波が出力されているのがわかります(**図B-1**のいちばん上とその次)．また，内部変数も同時に表示できるので，プログラムのデバッグに便利です．

すでに指摘したとおり，周辺モジュールのエミュレーションも行えます．ここで表示している例はSPIです．設定レジスタの各ビットの設定をチェックボックス形式で表示してくれます．もちろんクリックして書き換えることもできます．しかし，やはり重要なのは，このような見やすい形式で表示してくれるため，勘違いなどがないということです．組み込みマイコンの世界では，周辺モジュールの設定が多く，またややこしいため，迷路にはまり込んでしまうことがよくあります．初めて使用するマイコンであればなおさらです．その際，このように簡単に確認できるのは，非常に助かります．

さらに，UARTのウィンドウ(Serial #1)では入出力を行えます．今回のプログラムではオープニング・メッセージを表示させていますが，それはここに表示されます．また，周波数の変更などの指示をUART経由で行いますが，ここに入力することにより，プログラムに渡すことができます．したがって，エミュレータ上で完全にデバッグすることができるのです．

また，このデバッガには何らかのチェック機能も付いているようで，例えば配列の添え字の範囲を超えてアクセスしてしまったとき，エラーを検出してくれました．実際にCプログラムでこれを行うと，次の変数領域をそのままアクセスするだけで，通常は暴走してしまいます(実際，チップは確かに反応しなくなった)．こうなるとデバッグはかなり難しいのですが，このデバッグ機能は処理を停止してくれました．そのため，デバッグ作業ではおおいに助かりました(もしかすると，添え字の範囲の逸脱が大きすぎて，アドレス割り付けされていないところにアクセスし，たまたまエラーになったのかもしれないが…)．

こう言ってしまうとべたぼめのようですが，もちろん問題もありました．例えば古いバージョンではコンパイラに不ぐあいがあったり，ロジック・アナライザ機能に波形を取り込む際の名まえの指定のしかたがわかりにくかったり，そもそも周波数がずれていたり．これらは新しいバージョンでは修正されているので，ぜひ新しいバージョンを入手するようにしてください．ただし，新しいバージョンでも，UARTのウィンドウでは改行してくれないとか，波形表示できるのがグロー

ありませんでした．ただ，どこまで追従できるかというのはばねの形状によるようです．今回用いた材料（チタン0.1 mm板）の場合，550Hz以上の三角波に追従させようとするのであれば，ばねを何個か試すことが必要でした．550 Hz以下であれば，どのばねでも追従できました．また，ばねにより，振幅が大きくなる周波数，あるいは振動が安定する周波数も異なるようなので，やはりいくつか試すことをお勧めします．周波数は，UART経由でタイマの割り込みカウンタを操作し，1サイクル単位に調整できます．これにより，作成した系で安定して発振するポイントを探します．

実際の調整ですが，まず周波数を振って，高速方向に安定して発振する点を探します．次に，その周波数において十分なフレーム・レートが得られる点を，走査ステップを変えながら選びます．この時点で四角形が描けているはずです．なお，低速に振っているY方向の場合，ばねとソレノイドの間隔も微妙なので，同時に調整します．次に，オフセットを確認しやすい図形をダウンロードし，オフセットを調整します．ここまでできたら，あとは希望のビットマップをダウンロードするだけで絵が描けます．

● 実際の製作では部品選びや素材選びが重要

今回作成したレーザ・ディスプレイ装置とその表示結果を**写真2**に示します．**写真2（c）**のX方向，Y方向の走査周波数は，それぞれ787Hz，16.4Hzとしています．

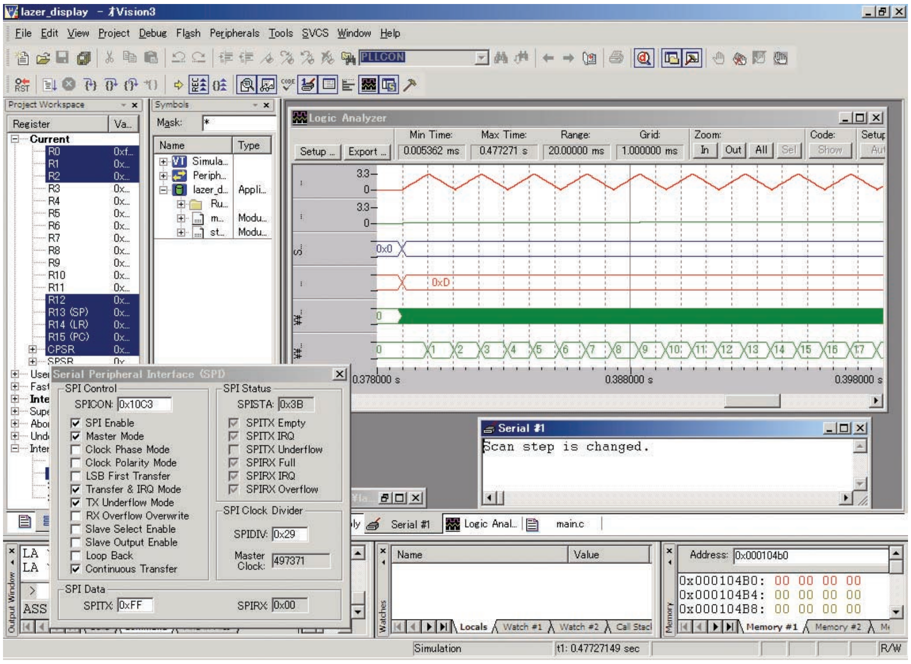
以下に，実際に作成するときのポイントを挙げます．

まず，レーザについては，できるだけ光束の細いものを使用することが必要です．ただ，これは買ってみなければわかりませんが…．また，最終的に光学系を組み立てることになるため，固定しやすい形のものがよいと思います．今回は，平型（チューインガム型）のものを使用しています．なお，レーザについてはそれなりに危険な光線であることを認識してください（p.59のコラム「レーザのクラス分けについて」を参照）．通常に使用する範囲では，クラス1

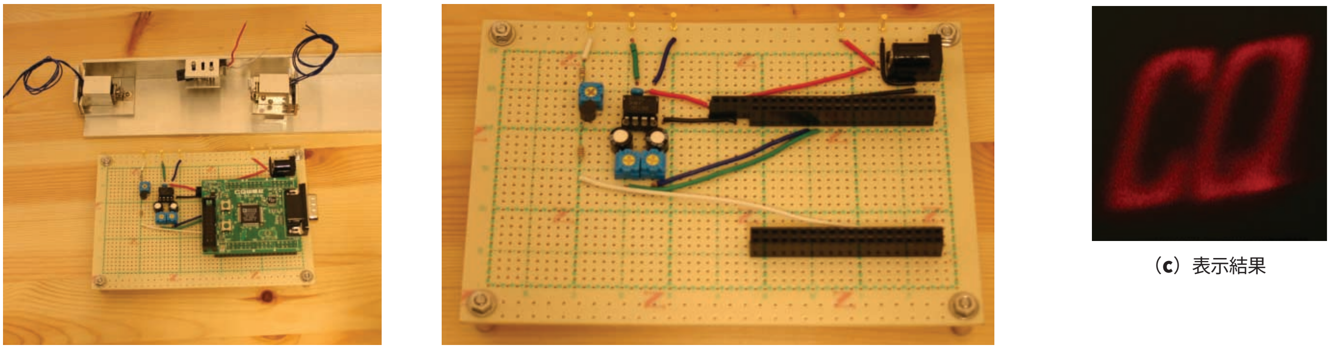
4

バル変数だけだとか，肝心の波形は変化点が画面内にないときに'0'になってしまうとか，いくつか問題はあります．しかし，それを考慮に入れても，とても使いやすいツールであることは確かです．上述の問題も，今後のバージョンアップによって修正されることを期待します．

初心者にとっては，UARTを動かすだけでもたいへんな組み込みプログラミングですが，このような開発環境があれば，チャレンジしやすいのではないでしょうか．



図B-1
デバッグ・セッション
デバッグ・セッションでは，周辺モジュールのレジスタを見たり，エミュレーションを行える．例えばUARTに対して信号を入力したり，出力ピンや内部変数を波形として観測できる．



（a）システム全体　　（b）作成した基板　　（c）表示結果

写真2　レーザ・ディスプレイ装置の外観と表示結果

(a)のようにARM7基板に簡単な回路を追加し，上のディスプレイ部を制御して描画した．基板と装置の間はクリップで結線している．(b)のように，部品点数は少ない．制御に必要な信号をマイコンから出力し，基板上の回路によって駆動能力を上げ，そのままディスプレイ部に入力する．(c)では，“ CQ ”の文字を表示している．ビットマップは256ビット×48ビット，フレーム速度は32.8フレーム/s．装置を設置した机上から3.5m離れた壁に描き出したもの．描画サイズは5cm角程度．

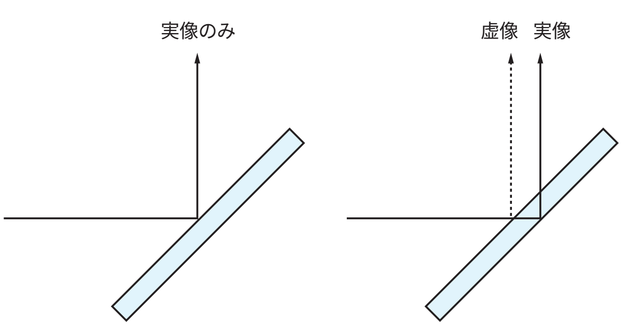
であれば問題なく安全です．今回使用したのもクラス1ですが，定格の40mAで十分すぎるほど明るかったことを付け加えておきます．

ばねの材料については，いろいろ試したのですが，なかなか高速に振れませんでした．試行錯誤の結果，できるだけ高速に振れる材料として，今回はチタン0.1mm板を用いました．特性としてはアルミでもほぼ同じなのですが，アルミの0.3mm板では堅すぎ，0.1mm，0.2mmでは柔らかすぎ，ちょうどいい感じになりませんでした．

理論的なことに簡単に触れておきます．ばねが自由振動をした場合，振動数は$\sqrt{k/m}$に比例することが知られています．ここで，ばねの特性を示しているkは，$\sqrt{E/\rho}$に比例します．Eはヤング率で，簡単に言えば引っ張ったときの堅さです．ρは密度です．これらは物質固有のものなので，材料ごとに調べることができます．なお，固有振動数なども，同じようにこの値に比例します．今回，手に入った金属板の中では，チタンがいちばんこの値が大きかったため，高速追従性が良いだろうと期待して使用しました．

ミラーについては，ガラスのミラーも試したのですが，最終的には塩化ビニル（塩ビ）ベースの表面反射ミラーを使用しています．これは，万華鏡用として売っていました．加工もしやすく，軽く，また表面反射であるため，虚像も出ません（**図4(a)**）．厳密には，塩ビ・ベースで，これをカッター・ナイフで刻んで加工しているため，やや平面性が悪いかもしれません．最近は光学用のミラーなどもインターネット通販で手に入るようなので，必要であれば入手してみてもいいと思います．

ソレノイドについては，実際にはもっと高電圧を印加するなどの方法により，パワーを上げることができます．もっとよい特性を出すためには，ばねにもう少し堅めの金属板を用い，ソレノイドに高電圧を印加するのが良いと思われます．こうすれば，ミラーやソレノイドそのものの質量の影響が小さくなるからです．ソレノイドの定格は「温度限界」で決まります．今回使用したソレノイドの定格は，平均消費電力1Wで内部抵抗36Ωということなので，三角波であれば10V程度まで印加しても問題ないはずです．今回は5Vの印加だったので，定格まで使用すると倍の力が発生すると期待できます．ただし，ソレノイドは本来このような高速往復運動に使用するものではなく，もっと低速な運



（a）表面反射ミラー　　（b）裏面反射ミラー

図4　表面反射ミラー

一般に使用されているミラーは「裏面反射ミラー」であり，今回のように斜めに使用するときにこれを使うと，表面で反射された一部の光線が虚像を構成してしまう．この結果，像がぼやけることになる．これに対して，レーザ光学系で一般に使用されているミラーは「表面反射ミラー」であり，虚像を構成させない．しかし，表面で反射するということは，鏡面が露出しているということでもあり，寿命は短くなる．

Column レーザのクラス分けについて

レーザは，その危険性でクラス分けされています．一般に販売されている半導体レーザの多くはクラス1で，出力は1mW以下です．通常は，この光量で十分です．秋葉原ではクラス3Bのレーザ（出力5mW）も普通に手に入りますが，使用する必然性はないでしょう．いずれにしても，レーザはそれなりに危険なので，取り扱いには注意してください．

参考までに，レーザについてのJISの安全基準（レーザのみ）について簡単にまとめてあったWikipedia(1)*から引用すると，以下のようになります．

- **クラス1**——100秒以内ならビームを直視しても安全
- **クラス2**——ビームを直視しても安全だが，まばたきが必要
- **クラス3A**——レンズなどの光学的手段を用いたビーム直視は不可
- **クラス3B**——ビーム直視は不可
- **クラス4**——ビーム直視はもちろん，反射拡散光も見ることは不可

参考・引用*文献

（1）Wikipedia，http://ja.wikipedia.org/wiki/

動に使用するものです．交流電流を流すことを想定して作られているかどうかは不明です．したがって，余裕を持って使用することをお勧めします．

光学系は，ミラー機構二つとレーザ・モジュールを角度をつけて固定する機構を，両面テープでアルミ台にくっつけています．これらの機構はたいした重さではないので，両面テープの接着力で十分固定できます．実際に光線をミラーにきっちり当てないと描画できないので，調整が必要ですが，これは意外と簡単に合わせることができました．

● アニメーションやスクロールも実現可能

このように，レーザ・ディスプレイ装置を試作してみた結果，多少ぼやけてはいますが，ビットマップを出力できました．材料を変えてみたり，あるいはD-A出力波形に補正を加えてみたりすることにより，もっと鮮明な画像を出力することも可能でしょう．金属加工に慣れていない方には，ねじの加工はちょっと難しいかもしれませんが，両面テープによる固定でも絵は描けます（時間方向には安定しないが…）．

半導体レーザも簡単に手に入るようになったことですし，試しにやってみるのもおもしろいのではないでしょうか．ビットマップはRAM上にフレーム・メモリとして持っているため，アニメーションやスクロールなども自由自在です．

たけうち・せいじ
設計エンジニア

<筆者プロフィール>
竹内誠二．久しぶりにはんだごてを手にして，電子工作を行ってみました．とはいっても，回路は部品点数が数点，プログラムはごくごく単純で，ほとんどの時間は機械工作でしたが…．はんだごてを握ったのは，二つ前の夏にパソコンのメイン・ボードを改造したとき以来ですね．